Fall 2015

Midterm Exam Report

Yeojin Kim *Advance Algorithm Programming*

1 Summary of the two methods

1.1 hedcuter method

Hedcuter method consists of two main steps which is slightly different version from Secord's method[2]:

1. Sample initial points $X = \{\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, \cdots, \mathbf{x_n}\}$
2. Loop while the displacements don't exceed maximum site displacement and the number of iterations doesn't exceed maximum number of iterations,
   (a) Map the image value at $\mathbf{x_i}$ to stipple level $t_{\mathbf{x_i}}$
   (b) Propagate the stipple level at $\mathbf{x_i}$ to its neighbor
   (c) Compute the centroids $\mathbf{C_i}$
   (d) Move each points $\mathbf{x_i}$ to its new centers of a cell $\mathbf{C_i}$

First of all, hedcuter samples n points that spread evenly in black area. The algorithm samples a uniformly distributed random point for whole area of given image and test if this random point has high probability based on gaussian distribution. If that is the case, it means that the value of the random point is close to black. The algorithm keeps this point as an initial point and this process is repeated until generating n points.

In the loop, the algorithm keeps mapping image using the location of a point(which corresponds to the coordinates of a pixel) as unique identification. This mapping image contains stipple level t at each points with the number of levels l. Unlike Secord's method[2], hedcuter method first computes and stores stipple level t only at given points using Eq. 1.

$$t_{\mathbf{x}} = \frac{l - I(\mathbf{x})}{l} \tag{1}$$

Next, it sorts the stipples with stipple level, x values, and y values in ascending order. Stipple level has a highest priority and y values has a lowest priority in sorting. Starting from a stipple which has the lowest stipple level, check the neighbors of the stipple. A stipple level t_{new} of a neighbor nb can be newly computed from the current stipple c based on Eq. 2.

$$t_{new} = t_c + t_{nb} \tag{2}$$

In case that a neighbor has a lower stipple level than stipple level it already has, hedcut method keeps new stipple level and add this neighbor as stipple. After propagation of stipple level through all pixels, hedcuter method collect pixels which is affected by given points, so called coverage of a site. When computing the new center of a stipple, it compute weighted average of stipple levels in coverage.

1.2 voronoi method

A flow of voronoi method basically follows Lloyd's method[1], which as follows :
when the centroid of a region is defined as

$$\mathbf{C_i} = \frac{\int_A \mathbf{x}\rho(\mathbf{x})dA}{\int_A \rho(\mathbf{x})dA}, \tag{3}$$

Algorithm 1 Lloyd's method

```
1: while generating point $\mathbf{x_i}$ not converged to centroids do
2:     Compute the Voronoi diagram of $\mathbf{x_i}$
3:     Compute the centroids $\mathbf{C_i}$ using equation (3)
4:     Move each generating point $\mathbf{x_i}$ to its centroids $\mathbf{C_i}$
5: end while
```

For more simple explanation of details, let's assume that we have a part of image(Fig. 1(a)) from squirrel's back. First, voronoi method samples points on the image randomly(Fig. 1(b)).

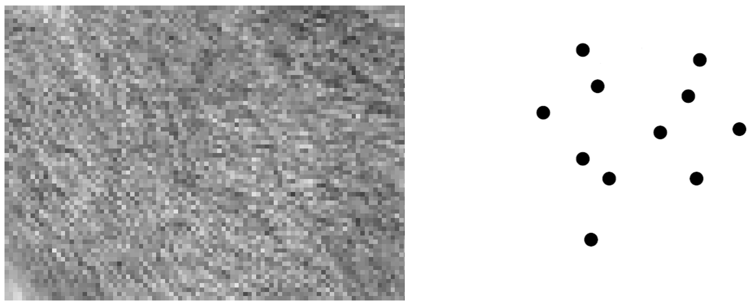
(a) (b)

Figure 1: A part of given image and sample points.

Next, voronoi method computes the voronoi region of given sample points(Fig. 2(a)). After creating voronoi diagram of given points, the algorithm computes the centroid of a cell iteratively looping through all these points. For example, with a given cell in Fig. 2(b), voronoi method calculates the line which is the extension of voronoi edge and is called clipping line(Fig. 2(c)). When two end points of voronoi edge are $\mathbf{x_1} = (x_1, y_1)$ and $\mathbf{x_2} = (x_2, y_2)$, the equation of clipping line is

$$\begin{gathered}(y - y_1) = \frac{(y_2 - y_1)}{(x_2 - x_1)}(x - x_1) \\ (y - y_1)(x_2 - x_1) = (y_2 - y_1)(x - x_1) \\ (y_2 - y_1)(x - x_1) - (y - y_1)(x_2 - x_1) = 0 \\ (y_2 - y_1)x - (x_2 - x_1)y + y_1(x_2 - x_1) - x_1(y_2 - y_1) = 0 \\ -(y_1 - y_2)x + (x_1 - x_2)y + x_1(y_1 - y_2) - y_1(x_1 - x_2) = 0\end{gathered} \tag{4}$$

$$\therefore ax + by + c = 0$$
$$where\ a = -(y_1 - y_2),\ b = (x_1 - x_2),\ and\ c = x_1(y_1 - y_2) - y_1(x_1 - x_2).$$

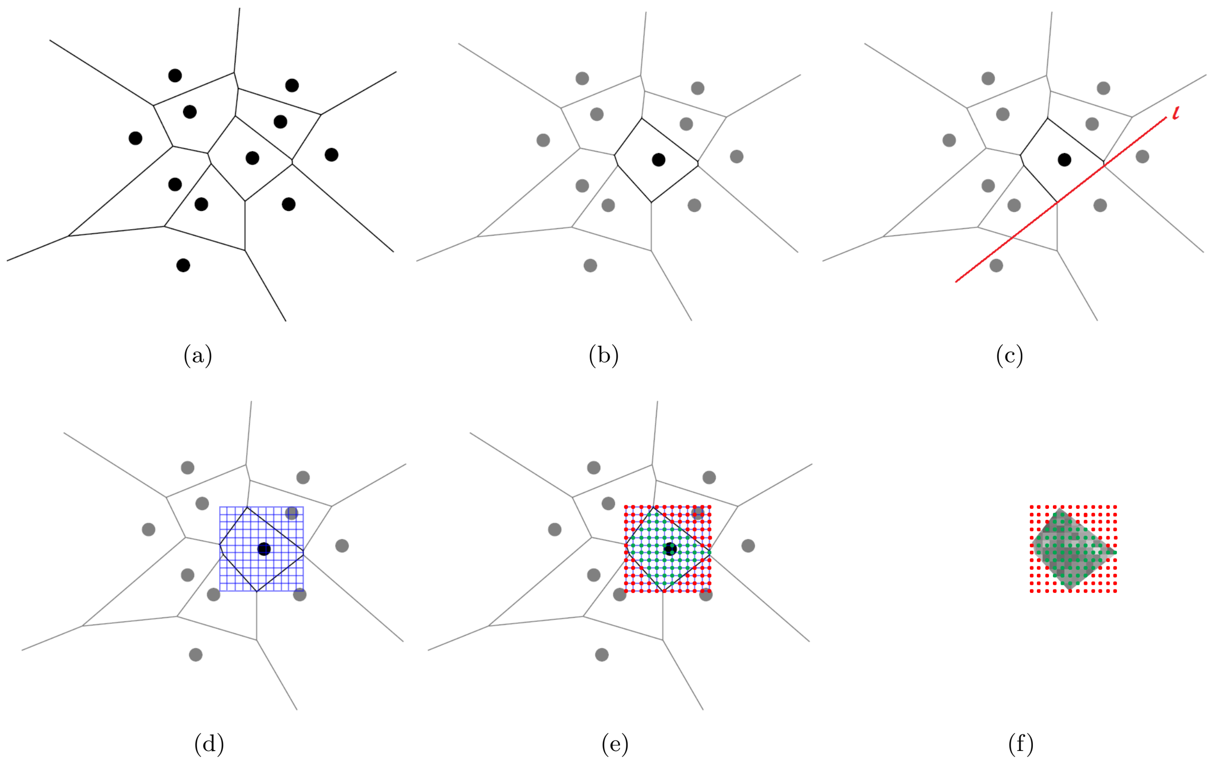

Figure 2: The progress of computing voronoi diagram and redistributing stipples.

In the same way, we can generate clipping lines for all voronoi edges in a cell.

To obtain the integration of density $\rho(\mathbf{x})$, voronoi method creates grid on a cell with the user-specified number of subpixels(Fig. 2(d)). Using the clipping lines(Eq. 4), we can test if a grid point is inside or outside of cell. If a grid point $\mathbf{x_g} = (x_g, y_g)$ satisfies $ax_g + by_g + c < 0$ for all the clipping lines, it locates inside of a cell. The result of test is shown in Fig. 2(e). The red points mean outside and the green points mean inside.
When grid points are inside, the algorithm obtains $\int_A \rho(\mathbf{x})dA$ and $\int_A \mathbf{x}\rho(\mathbf{x})dA$ with intensity $I(\mathbf{x_g})$ which has corresponding location to grid points in grayscale image(Fig. 2(f), Eq. 5).

$$\begin{aligned}\int_A \rho(\mathbf{x})dA &= \int I(\mathbf{x_g}) \\ \int_A \mathbf{x}\rho(\mathbf{x})dA &= \int \mathbf{x}I(\mathbf{x_g}) = \int (x,y)I(\mathbf{x_g}).\end{aligned} \tag{5}$$

Substituting the results of Eq. 5 to Eq. 3, we can generate new centroid $\mathbf{C_i} = (x_c, y_c)$, which satisfies that $x_c = \frac{\int x \cdot I(\mathbf{x_g})}{\int I(\mathbf{x_g})}$ and $y_c = \frac{\int y \cdot I(\mathbf{x_g})}{\int I(\mathbf{x_g})}$.This new centroids will be fed as sites into creating voronoi diagram step. Voronoi method repeats creating voronoi diagram and computing the centroid repeatedly until the average of displacement between points $\mathbf{x_i}$ and centroids $\mathbf{C_i}$ is small enough.

2 Comparison of the two methods

3 Improvement of hedcuter method

References

[1] Atsuyuki Okabe, Barry Boots, and Kokichi Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams.* John Wiley & Sons, Inc., New York, NY, USA, 1992. 1.2

[2] Andrian Secord. Weighted voronoi stippling. In *2nd International Symposium on Non-Photorealistic Animation and Rendering (NPAR'02)*, pages 37–43, Annecy, France, June 3-5 2002. 1.1, 1.1